

# Creación de una aplicación web de alta disponibilidad y escalabilidad



**Autor: Serhiy Holovasin**

**Tutora: María Jesús Sánchez García**

**IES Infanta Elena (Galapagar-Madrid)**

# ÍNDICE DE CONTENIDOS:

<b>Creación de una aplicación web de alta disponibilidad y escalabilidad .....</b>	<b>1</b>
<b>Situación .....</b>	<b>3</b>
<b>Fase 1: planificar el diseño y estimar los costos .....</b>	<b>5</b>
Tarea 1: crear un diagrama de la arquitectura .....	6
Tarea 2: desarrollar una estimación de costos.....	7
<b>Fase 2: creación de una aplicación web funcional básica .....</b>	<b>9</b>
Tarea 1: crear una red virtual.....	9
Tarea 2: crear una máquina virtual .....	11
Tarea 3: probar la implementación .....	15
<b>Fase 3: desacoplamiento de los componentes de la aplicación .....</b>	<b>16</b>
Tarea 1: modificar la configuración de la VPC .....	16
Tarea 2: creación y configuración de la base de datos de Amazon RDS .....	16
Tarea 3: configuración del entorno de desarrollo.....	19
Tarea 4: aprovisionamiento del Secrets Manager .....	22
Tarea 5: aprovisionamiento de una nueva instancia para el servidor web....	23
Tarea 6: migrar la base de datos .....	24
Tarea 7: probar la aplicación .....	26
<b>Fase 4: implementar la alta disponibilidad y escalabilidad .....</b>	<b>28</b>
Tarea 2: implementación de Amazon EC2 Auto Scaling.....	31
Tarea 3: acceder a la aplicación .....	34
Tarea 4: prueba de carga de la aplicación .....	35
<b>Evaluación del Cumplimiento de los Objetivos .....</b>	<b>38</b>
<b>Conclusión .....</b>	<b>39</b>

# Creación de una aplicación web de alta disponibilidad y escalabilidad

## Situación

La Universidad Ejemplo se prepara para el nuevo año escolar. El Departamento de Admisiones recibió reclamos de que su aplicación web para los expedientes de los estudiantes es lenta o no está disponible durante el período de admisiones más alto debido al elevado número de consultas.

Usted es un ingeniero de la nube. Su gerente le pidió que cree una prueba de concepto (POC) para alojar la aplicación web en la nube de AWS. Su gerente desea que diseñe e implemente una nueva arquitectura de hospedaje que mejore la experiencia de los usuarios de la aplicación web. Usted es responsable de construir la infraestructura para alojar la aplicación web de registros de estudiantes en la nube.

Su desafío consiste en planificar, diseñar, crear e implementar la aplicación web en la nube de AWS de forma coherente con las prácticas recomendadas del Marco de AWS Well-Architected. Durante el período de admisiones más alto, la aplicación debe admitir miles de usuarios y ser de alta disponibilidad, ser escalable, tener un balanceo de carga, ser segura y ofrecer un alto rendimiento.

En la imagen siguiente se muestra un ejemplo de la aplicación web para el registros de estudiantes. En el sitio se muestran los expedientes de los estudiantes que solicitaron su admisión en la universidad. Los usuarios pueden ver, añadir, borrar y modificar los registros de los estudiantes.

Name	Address	City	State	Email	Phone
John Doe	Example Address	Example City	example State	example@example.com	9009009009

Add a new student

## Requisitos de la solución

La solución debe cumplir los siguientes requisitos:

- Funcional:** la solución cumple los requisitos funcionales, como la posibilidad de ver, añadir, eliminar o modificar los expedientes de los estudiantes, sin ningún retraso perceptible.

- **Balanceo de carga:** la solución puede equilibrar adecuadamente el tráfico de usuarios para evitar la sobrecarga o infrautilización de los recursos.
- **Escalable:** la solución está diseñada para adaptarse a la demanda de la aplicación.
- **Alta disponibilidad:** La solución está diseñada para tener un tiempo de inactividad limitado cuando un servidor web deja de estar disponible.
- **Segura:**
  - La base de datos está protegida y no se puede acceder a ella directamente desde redes públicas.
  - Solo se puede acceder a los servidores web y a la base de datos a través de los puertos adecuados.
  - La aplicación web es accesible a través de internet.
  - Las credenciales de la base de datos no están codificadas en la aplicación web.
- **Optimización de costos:** la solución está diseñada para mantener los costos bajos.
- **Alto rendimiento:** las operaciones rutinarias (ver, añadir, borrar o modificar registros) se realizan sin un retraso perceptible bajo cargas normales, variables y máximas.

## **Supuestos**

Este proyecto se construirá en un entorno de laboratorio controlado que tiene restricciones en cuanto a servicios, funciones y presupuesto. Tenga en cuenta lo siguiente para el proyecto:

- La aplicación se implementa en una región de AWS (no es necesario que la solución sea multirregional).
- El sitio web no necesita estar disponible sobre HTTPS o un dominio personalizado.
- La solución se implementa en máquinas *Ubuntu* utilizando el código JavaScript que se proporciona.
- Utilice el código JavaScript tal y como está escrito a menos que las instrucciones le indiquen específicamente que cambie el código.
- La solución utiliza servicios y funciones dentro de las restricciones del entorno de laboratorio.
- La base de datos solo está alojada en una única zona de disponibilidad.
- El sitio web es de acceso público sin autenticación.
- La estimación del costo es aproximada.

## **Descargo de responsabilidad:**

Una de las mejores prácticas recomendadas de seguridad es permitir el acceso al sitio web a través de la red de la universidad y la autenticación. Sin embargo, debido a que está construyendo esta aplicación como una POC, esas funciones están fuera del alcance de este proyecto. Se lo anima a implementar esta funcionalidad adicional.

## **Solución:**

Para implementar el acceso al sitio web solo a través de la red de la universidad y la autenticación podemos seguir siguientes pasos:

### *Acceso Solo a Través de la Red de la Universidad*

#### 1. Restricción de IP en los Grupos de Seguridad:

Configurar los Grupos de Seguridad de nuestras instancias EC2 para que solo permitan el tráfico entrante desde las direcciones IP de la red de la universidad. Esto se hace editando las reglas de entrada del grupo de seguridad y especificando las IP de origen.

#### 2. Usar AWS WAF (Web Application Firewall):

Podemos utilizar AWS WAF para agregar una capa adicional de seguridad y definir reglas que permitan el acceso solo desde las IP de la universidad. Crear una ACL web en AWS WAF y definir una regla basada en IP.

### *Implementar Autenticación*

#### 1. Configurar Amazon Cognito:

Amazon Cognito proporciona autenticación, autorización y gestión de usuarios para aplicaciones web y móviles. Crear un usuario de grupo y una aplicación en Amazon Cognito.

#### 2. Integrar Cognito con nuestra Aplicación Web:

Modificar nuestra aplicación web para utilizar Amazon Cognito para la autenticación. Incluir la biblioteca de Amazon Cognito en la aplicación y configurar la autenticación.

Estos pasos nos ayudarán a mejorar la seguridad de la aplicación web restringiendo el acceso a la red de la universidad y añadiendo autenticación de usuario. Si bien estas prácticas están fuera del alcance del POC, son esenciales para una implementación real en producción.

## **Fase 1: planificar el diseño y estimar los costos**

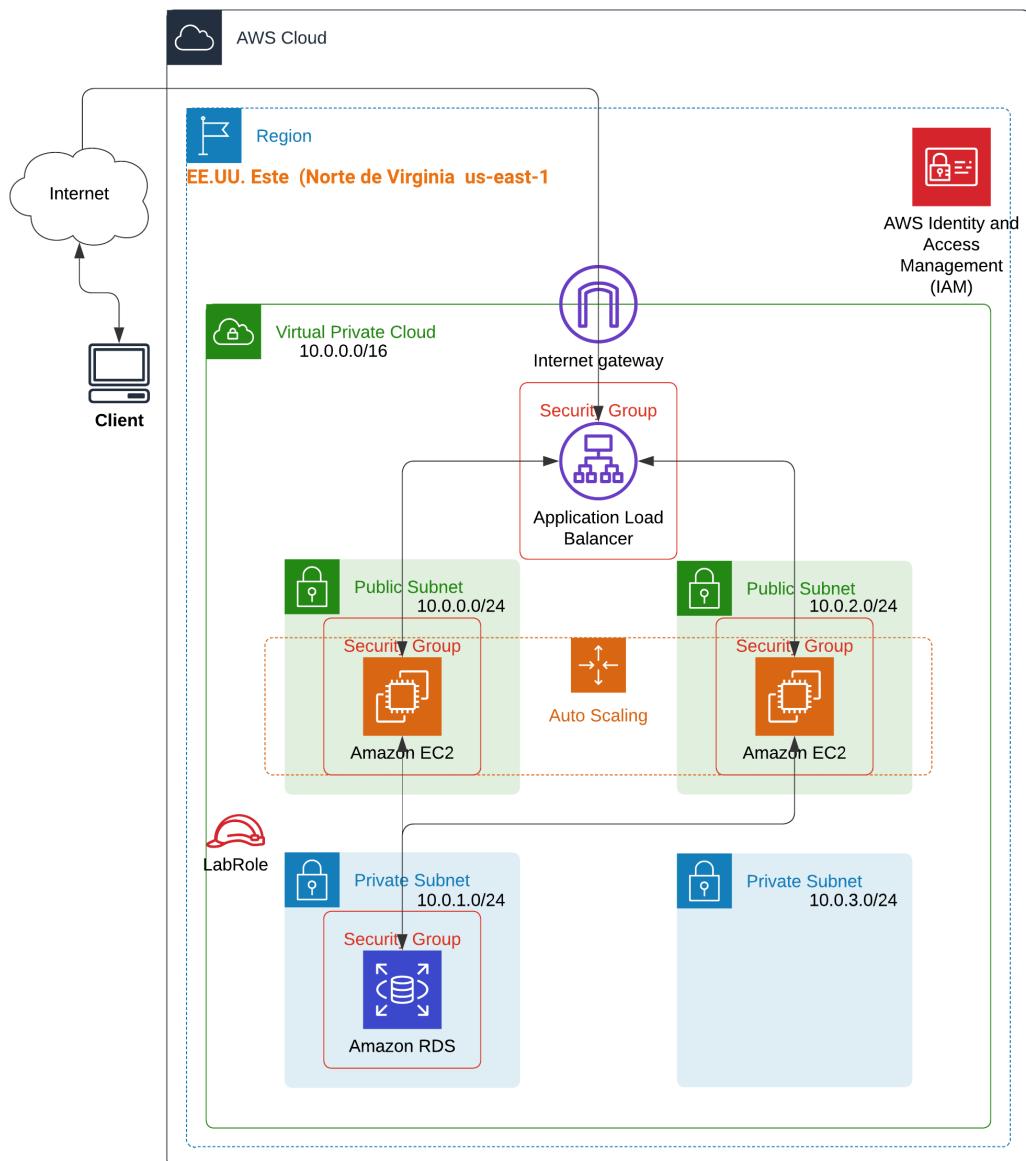
*En esta fase, planificará el diseño de su arquitectura. Primero, creará un diagrama de la arquitectura.*

*A continuación, estimará el costo de la solución propuesta y presentará la estimación a su instructor. Un primer paso importante de cualquier solución es planificar el diseño y estimar los costos. Revise los diversos componentes de la arquitectura para ajustar los costos estimados según*

se necesite. El costo es un factor importante a la hora de construir una solución, ya que puede determinar los componentes y el patrón de arquitectura que hay que utilizar.

## Tarea 1: crear un diagrama de la arquitectura

Cree un diagrama de arquitectura para ilustrar lo que planifica crear. Considere cómo cumplirá cada requisito de la solución.



Los usuarios acceden a la aplicación web desde sus navegadores a través de Internet. Las solicitudes de los clientes son dirigidas a la URL del Application Load Balancer (ALB), que actúa como el punto de entrada a la arquitectura. El ALB se encarga de recibir las solicitudes entrantes y distribuirlas equitativamente entre las instancias EC2 desplegadas en las subredes públicas. Este balanceo de carga asegura que ninguna instancia EC2 se sobrecargue, manteniendo la alta disponibilidad y el rendimiento de la aplicación. Un grupo de seguridad asociado al ALB permite el tráfico entrante en el puerto 80 (HTTP).

Las instancias EC2 alojan la aplicación web y procesan las solicitudes recibidas del ALB. Estas instancias ejecutan el código de la aplicación, gestionando las operaciones solicitadas por los usuarios, como ver, añadir, eliminar o modificar registros de estudiantes. Las instancias EC2 están configuradas en un Auto Scaling Group, que ajusta automáticamente el número de instancias en función de la demanda, asegurando que siempre haya suficientes recursos disponibles para manejar el tráfico de la aplicación. Un grupo de seguridad para las instancias EC2 permite el tráfico entrante solo desde el ALB en el puerto 80 (HTTP) y el tráfico SSH solo desde direcciones IP específicas para administración.

Para las operaciones que requieren acceso a datos persistentes, las instancias EC2 se comunican con una base de datos MySQL alojada en Amazon RDS. La base de datos RDS está desplegada en subredes privadas para mayor seguridad y solo permite conexiones desde las instancias EC2 autorizadas. AWS Secrets Manager se utiliza para gestionar las credenciales de la base de datos de manera segura, proporcionando a las instancias EC2 acceso seguro a los secretos necesarios para conectarse a RDS. El grupo de seguridad para RDS permite el tráfico entrante solo desde los grupos de seguridad asignados a las instancias EC2 en el puerto 3306 (MySQL).

Además, Amazon Cognito se utiliza para la autenticación y gestión de usuarios, aunque su implementación completa queda fuera del alcance del POC. Cognito facilita la creación de un sistema de autenticación robusto y seguro para la aplicación web, gestionando las credenciales de los usuarios y proporcionando características avanzadas de seguridad.

Finalmente, AWS Identity and Access Management (IAM) gestiona los permisos y roles necesarios para que los servicios interactúen de manera segura. Roles como LabRole se asignan a las instancias EC2 para permitirles acceder a los secretos gestionados por AWS Secrets Manager y otros recursos necesarios. Este enfoque asegura que cada componente de la arquitectura tenga los permisos mínimos necesarios para funcionar, siguiendo las mejores prácticas de seguridad de AWS.

## Referencias

- [Íconos de arquitectura de AWS](#): este sitio ofrece herramientas para dibujar diagramas de arquitectura de AWS.
- [Diagramas de arquitectura de referencia de AWS](#): este sitio ofrece diagramas de arquitectura de referencia para diversos casos prácticos.

---

## Tarea 2: desarrollar una estimación de costos

*Elabore una estimación de costos que muestre el costo de ejecutar la solución en la región us-east-1 durante 12 meses. Use la [AWS Pricing Calculator](#) para esta estimación.*

### *Resumen de la Estimación de Costos AWS*

Esta es una estimación y puede variar dependiendo de varios factores. Si se implementan características adicionales como NAT Gateways, AWS WAF para limitar el acceso o autenticación Serhiy Holovasin

avanzada con Amazon Cognito, los costos pueden aumentar. Es importante monitorear y ajustar la configuración para optimizar los costos de acuerdo a las necesidades reales de la aplicación.

The screenshot shows the AWS Pricing Calculator interface. At the top, it displays a summary of the estimated costs: Costo inicial (0,00 USD), Costo mensual (50,00 USD), and Costo total de 12 months (600,00 USD, including the initial cost). On the right, there are buttons for 'Exportar' and 'Compartir'. Below the summary, there's a section titled 'Comenzar con AWS' with links to 'Comience de forma gratuita' and 'Comuníquese con el departamento de ventas'. The main area is titled 'My Estimate' and lists the services included in the estimate with their respective costs and descriptions. Services listed include Amazon Virtual Private Cloud (VPC), Elastic Load Balancing, Amazon EC2, Amazon RDS for MySQL, and Amazon Cognito.

Nombre del servicio	Estado	Costo inicial	Costo mensual	Descripción	Región	Resumen de la configuración
Amazon Virtual Private Cloud (VPC)	-	0,00 USD	0,00 USD	-	Este de EE. UU. ...	Días laborables al mes (22), Número de conexiones de Site-to-Site VPN (0), Número de asociaciones de subred ...
Elastic Load Balancing	-	0,00 USD	16,47 USD	-	Este de EE. UU. ...	Número de balanceadores de carga de aplicaciones (1)
Amazon EC2	-	0,00 USD	18,58 USD	-	Este de EE. UU. ...	Tenencia (Instancias compartidas), Sistema operativo (Linux), Carga de trabajo (Daily (Días de carga de trabajo ...
Amazon RDS for MySQL	-	0,00 USD	14,71 USD	-	Este de EE. UU. ...	Cantidad de almacenamiento (20 GB), Almacenamiento para cada instancia RDS (SSD de uso general (gp2)), N...
Amazon Cognito	-	0,00 USD	0,25 USD	-	Este de EE. UU. ...	Tasa de optimización para solicitudes de tokens (0), Tasa de optimización para clientes de aplicaciones (0), Car...

Para el proyecto de implementación de una aplicación web de alta disponibilidad y escalabilidad en AWS, se ha estimado un costo mensual de \$50 USD, lo que resulta en un costo total de \$600 USD para 12 meses. A continuación, se detallan los componentes de la estimación:

#### *Amazon Virtual Private Cloud (VPC)*

El uso de Amazon VPC no genera costos adicionales en esta estimación. Se considera que no se aplicarán configuraciones específicas que incurran en costos, como conexiones VPN o direcciones IPv4 públicas en uso.

#### *Elastic Load Balancing (ELB)*

El balanceador de carga de aplicaciones (Application Load Balancer) se estima en \$16.47 USD al mes. Este componente es crucial para distribuir el tráfico entre las instancias EC2, asegurando así la alta disponibilidad y escalabilidad de la aplicación.

#### *Amazon EC2*

El costo mensual estimado para las instancias EC2 es de \$18.58 USD. Se ha configurado una instancia t2.micro de Linux, operando con una estrategia de precios sin pago inicial por un año. Esta configuración cubre las necesidades básicas de cómputo de la aplicación web.

#### *Amazon RDS for MySQL*

La base de datos MySQL en Amazon RDS está estimada en \$14.71 USD al mes. La configuración incluye 20 GB de almacenamiento en SSD de uso general y una instancia db.t3.micro en una sola zona de disponibilidad. Esta configuración es adecuada para manejar las operaciones de la base de datos de la aplicación.

### *Amazon Cognito*

Amazon Cognito, utilizado para la gestión de usuarios y autenticación, tiene un costo estimado de \$0.25 USD al mes. Esta estimación incluye características de seguridad avanzadas y se basa en la suposición de 5 usuarios activos mensuales.

### *AWS Secrets Manager*

AWS Secrets Manager, utilizado para gestionar las credenciales de manera segura, tiene un costo estimado de aproximadamente \$1 USD al mes. Este costo incluye el almacenamiento de un secreto y hasta 100,000 llamadas mensuales para acceder a los secretos.

### *Resumen de Costos*

En total, los costos mensuales estimados suman \$50 USD. Este cálculo incluye el balanceador de carga, instancias EC2, base de datos RDS, gestión de usuarios con Cognito y AWS Secrets Manager. La estimación refleja una solución eficiente y escalable que cumple con los requisitos del proyecto, manteniendo los costos controlados.

### **Referencias**

- [¿Qué es AWS Pricing Calculator?](#)
- [Plantilla de presentación PowerPoint](#)

## **Fase 2: creación de una aplicación web funcional básica**

*En esta fase, empezará a construir la solución. El objetivo de esta fase es tener una aplicación web funcional que funcione en una sola máquina virtual en una red virtual que cree. Al final de esta fase, tendrá una POC para demostrar el alojamiento de la aplicación en la nube de AWS. A continuación, puede construir sobre su trabajo en fases posteriores.*

---

### Tarea 1: crear una red virtual

Para comenzar con la configuración de nuestra arquitectura en AWS, el primer paso es crear una red virtual privada (VPC) que contenga nuestras subredes y recursos de red. Esta VPC será el entorno donde desplegaremos todos los componentes necesarios para nuestra aplicación web.

Primero, accedemos a la consola de AWS y seleccionamos “VPC” en el menú de servicios. Dentro de la sección “Your VPCs”, seleccionamos la opción “Create VPC”. Proporcionamos el nombre “proyecto-vlc” para la VPC y definimos una dirección CIDR, por ejemplo, 10.0.0.0/16, que permitirá un amplio rango de direcciones IP internas.

A continuación, necesitamos añadir subredes dentro de esta VPC para separar los componentes públicos y privados de nuestra arquitectura. Creamos dos subredes públicas y dos subredes privadas.

Asignamos las direcciones CIDR a las subredes de la siguiente manera:

- Subred pública 1: 10.0.0.0/24
- Subred pública 2: 10.0.2.0/24
- Subred privada 1: 10.0.1.0/24
- Subred privada 2: 10.0.3.0/24

El asistente de configuración creará automáticamente una Gateway de Internet y asociará esta Gateway a la VPC. Además actualizará las tablas de enruteamiento correspondientes para permitir el tráfico de entrada y salida. La tabla de enruteamiento pública se configurará para dirigir el tráfico de 0.0.0.0/0 hacia la Gateway de Internet, permitiendo que las instancias en las subredes públicas tengan acceso a Internet. La tabla de enruteamiento privada asegurará que las instancias en las subredes privadas no tengan acceso directo a Internet, proporcionando una capa adicional de seguridad.

VPC > Sus VPC > Crear VPC > Crear recursos de VPC

## Flujo de trabajo de creación de VPC

✓ Correcto

▼ Detalles

- ✓ Crear VPC: vpc-05cc22f6c0fa292fd [ ]
- ✓ Habilitar nombres de host DNS
- ✓ Habilitar la resolución de DNS
- ✓ Verificar la creación de una VPC: vpc-05cc22f6c0fa292fd [ ]
- ✓ Crear punto de enlace de S3: vpce-076e186be8e43887a [ ]
- ✓ Crear subred: subnet-08d923321893fb324 [ ]
- ✓ Crear subred: subnet-02b3f047da3518296 [ ]
- ✓ Crear subred: subnet-0d1c9c8d2e62e1f24 [ ]
- ✓ Crear subred: subnet-08396fbdd18f63116 [ ]
- ✓ Crear una gateway de Internet: igw-02a05cf18be9d18ca [ ]
- ✓ Adjuntar gateway de Internet a la VPC
- ✓ Crear tabla de enruteamiento: rtb-06f0669722b3950b9 [ ]
- ✓ Crear ruta
- ✓ Asociar tabla de enruteamiento
- ✓ Asociar tabla de enruteamiento
- ✓ Crear tabla de enruteamiento: rtb-01e0c6089287999a1 [ ]
- ✓ Asociar tabla de enruteamiento
- ✓ Crear tabla de enruteamiento: rtb-0ca9fd475c094c05d [ ]
- ✓ Asociar tabla de enruteamiento
- ✓ Verificando la creación de la tabla de enruteamiento
- ✓ Asociar el punto de conexión de S3 con tablas de enruteamiento de subred privada: vpce-076e186be8e43887a [ ]

Ver VPC

Con estos pasos, hemos configurado rápidamente una red virtual privada en AWS utilizando el asistente de configuración, que crea las subredes necesarias y la conectividad adecuada de manera automática. Esto establece una base sólida para desplegar los componentes de la aplicación en las siguientes tareas.

The screenshot shows the AWS VPC console interface. At the top, it displays the VPC ID: vpc-05cc22f6c0fa292fd and the name: proyecto-vpc. Below this, there are two main sections: 'Detalles' and 'Mapa de recursos'.

**Detalles:**

ID de la VPC vpc-05cc22f6c0fa292fd	Estado Available	Nombres de host de DNS Habilitado	Resolución de DNS Habilitado
Tenencia Default	Conjunto de opciones de DHCP dopt-032d2d832483ac0f9	Tabla de enrutamiento principal rtb-0b4480b90c80cfe60	ACL de red principal acl-0aa84274cbfeba185
VPC predeterminada No	CIDR IPv4 10.0.0.0/16	Grupo IPv6 -	CIDR IPv6 (grupo de bordes de red) -
Métricas de uso de direcciones de red Desactivado	Grupos de reglas del firewall de DNS de Route 53 Resolver -	ID de propietario 951516663729	

**Mapa de recursos:**

- VPC:** Mostrar detalles. Su red virtual de AWS: proyecto-vpc.
- Subredes (4):**
  - us-east-1a:** proyecto-subnet-public1-us-east-1a (highlighted in orange), proyecto-subnet-private1-us-east-1a...
  - us-east-1b:** proyecto-subnet-public2-us-east-1b (highlighted in orange), proyecto-subnet-private2-us-east-1a...
- Tablas de enrutamiento (4):**
  - projecto-rtb-public (highlighted in orange)
  - rtb-0b4480b90c80cfe60
  - projecto-rtb-private2-us-east-1b
  - projecto-rtb-private1-us-east-1a
- Conexiones de red (2):**
  - projecto-igw:** Rutas de Internet a 2 subredes públicas, 0 ruta(s) de subredes privadas a Internet.
  - projecto-vpce-s3

## Tarea 2: crear una máquina virtual

*Cree una máquina virtual en la nube para alojar la aplicación web.*

### Consejos:

- Use un servicio de cómputo como Amazon Elastic Compute Cloud (Amazon EC2).
- Utilice la última imagen de máquina de Amazon (AMI) de Ubuntu.

Ahora que hemos configurado nuestra red virtual privada, el siguiente paso es crear una instancia de máquina virtual (EC2) que alojará la aplicación web. Esta instancia EC2 será la base sobre la cual desplegaremos el código de la aplicación y otros componentes necesarios.

Primero, accedemos a la consola de Amazon EC2 desde el menú de servicios de AWS. Seleccionamos la opción “Launch Instance” para comenzar el proceso de configuración de una nueva instancia EC2. En la primera pantalla, seleccionamos una Amazon Machine Image (AMI) de Ubuntu. Ubuntu es una distribución de Linux que es ampliamente utilizada y bien soportada en AWS.

Después de seleccionar la AMI, elegimos el tipo de instancia t2.micro. Esta instancia es elegible para el nivel gratuito de AWS, lo cual es beneficioso para fines de desarrollo y pruebas. Además, es suficiente para manejar una carga básica de trabajo de la aplicación web en esta fase del proyecto.

## ▼ Imágenes de aplicaciones y sistemas operativos (Imagen de máquina de Amazon)

Información

Una AMI es una plantilla que contiene la configuración de software (sistema operativo, servidor de aplicaciones y aplicaciones) necesaria para lanzar la instancia. Busque o examine las AMI si no ve lo que busca a continuación.

 Busque en nuestro catálogo completo que incluye miles de imágenes de sistemas operativos y aplicaciones

### Inicio rápido



Buscar más AMI

Inclusión de AMI de AWS, Marketplace y la comunidad

### Imágenes de máquina de Amazon (AMI)

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type

ami-04b70fa74e45c3917 (64 bits (x86)) / ami-0eac975a54dfee8cb (64 bits (Arm))  
Virtualización: hvm Activado para ENA: true Tipo de dispositivo raíz: ebs

Apto para la capa gratuita

#### Descripción

Canonical, Ubuntu, 24.04 LTS, amd64 noble image build on 2024-04-23

#### Arquitectura

64 bits (x86) ▾

#### ID de AMI

ami-04b70fa74e45c3917

Proveedor verificado

## ▼ Tipo de instancia [Información](#) | [Obtener asesoramiento](#)

### Tipo de instancia

t2.micro

Apto para la capa gratuita

Familia: t2 1 vCPU 1 GiB Memoria Generación actual: true  
Bajo demanda Windows base precios: 0.0162 USD por hora  
Bajo demanda SUSE base precios: 0.0116 USD por hora  
Bajo demanda RHEL base precios: 0.0716 USD por hora  
Bajo demanda Linux base precios: 0.0116 USD por hora

Todas las generaciones

[Comparar tipos de instancias](#)

[Se aplican costos adicionales a las AMI con software preinstalado](#)

En la siguiente pantalla, configura la instancia para estar dentro de una de las subredes públicas de la VPC que creaste en la tarea anterior. Asegúrate de que la opción “Auto-assign Public IP” esté habilitada para que la instancia reciba una dirección IP pública, lo que permitirá que sea accesible desde Internet.

**▼ Configuraciones de red [Información](#)**

VPC : **obligatorio** | [Información](#)

vpc-05cc22f6c0fa292fd (proyecto-vpc)  
10.0.0.0/16

Subred | [Información](#)

subnet-02b3f047da3518296 proyecto-subnet-public2-us-east-1b  
VPC: vpc-05cc22f6c0fa292fd Propietario: 951516663729  
Zona de disponibilidad: us-east-1b Direcciones IP disponibles: 251 CIDR: 10.0.2.0/24

[Crear nueva subred](#)

Asignar automáticamente la IP pública | [Información](#)

Habilitar

**Se aplican cargos adicionales** cuando no se cumplen los límites del **nivel gratuito**

Firewall (grupos de seguridad) | [Información](#)

Un grupo de seguridad es un conjunto de reglas de firewall que controlan el tráfico de la instancia. Agregue reglas para permitir que un tráfico específico llegue a la instancia.

Crear grupo de seguridad  Seleccionar un grupo de seguridad existente

Grupos de seguridad comunes [Información](#)

Seleccionar grupos de seguridad

webserver-sg sg-053d95874181d9a0a X  
VPC: vpc-05cc22f6c0fa292fd

Compare reglas de grupo de seguridad

Los grupos de seguridad que agrega o elimine aquí se agregarán a todas las interfaces de red o se eliminarán de ellas.

► Configuración de red avanzada

EC2 > Grupos de seguridad > sg-053d95874181d9a0a - webserver-sg

Actions ▾

**Detalles**

Nombre del grupo de seguridad <input type="checkbox"/> webserver-sg	ID del grupo de seguridad <input type="checkbox"/> sg-053d95874181d9a0a	Descripción <input type="checkbox"/> Permite el acceso HTTP	ID de la VPC <input type="checkbox"/> vpc-05cc22f6c0fa292fd
Propietario <input type="checkbox"/> 951516663729	Número de reglas de entrada 2 Entradas de permiso	Número de reglas de salida 1 Entrada de permiso	

[Reglas de entrada](#) [Reglas de salida](#) [Etiquetas](#)

**Reglas de entrada (2)**

Reglas de entrada (2)		<input type="button"/> Administrar etiquetas	<input type="button"/> Editar reglas de entrada				
<input type="text"/> Buscar		< 1 >	①				
Name	ID de la regla del grupo	Versión de IP	Tipo	Protocolo	Intervalo de puertos	Origen	Descripción
<input type="checkbox"/> -	sgr-0a8b9893bd027e...	IPv4	SSH	TCP	22	0.0.0.0/0	Permita conexión SSH en el puerto...
<input type="checkbox"/> -	sgr-0c512838a5b1ab1...	IPv4	HTTP	TCP	80	0.0.0.0/0	Habilite el puerto 80

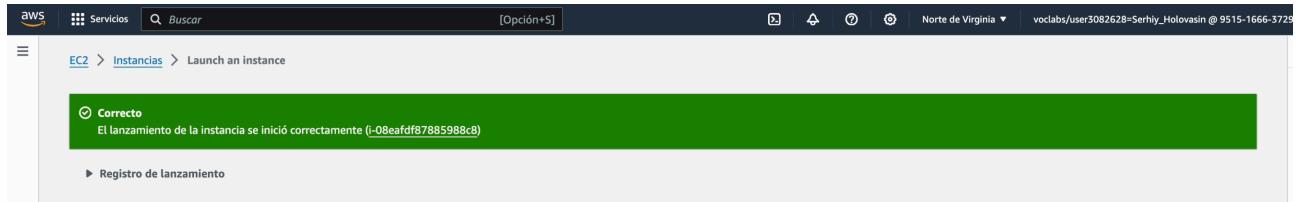
Configurar los grupos de seguridad es crucial para la seguridad y funcionalidad de la instancia EC2. Creamos un nuevo grupo de seguridad o en mi caso seleccionamos uno existente, lo he creado antes, que permita tráfico entrante en los puertos 80 (HTTP) y 22 (SSH). El puerto 80 permitirá el acceso web a la aplicación, mientras que el puerto 22 permitirá conexiones SSH para administración y configuración de la instancia. Sería aconsejable asegurarnos de restringir el acceso SSH a una dirección IP específica para mejorar la seguridad, pero como estamos en un laboratorio lo dejamos con 0.0.0.0/0.

También, durante el proceso de configuración, se nos pedirá que seleccionemos o creamos un par de claves. Este par de claves será necesario para conectarte a la instancia EC2 de manera segura.

```
#!/bin/bash -xe
apt update -y
apt install nodejs unzip wget npm mysql-server -y
# wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACCAP1-1-DEV/code.zip -P /home/ubuntu
wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACCAP1-1-79581/1-lab-capstone-project-1/code.zip -P /home/ubuntu
cd /home/ubuntu
unzip code.zip -x "resources/codebase_partner/node_modules/*"
cd resources/codebase_partner
npm install aws aws-sdk
mysql -u root -e "CREATE USER 'nodeapp' IDENTIFIED WITH mysql_native_password BY 'student12'";
mysql -u root -e "GRANT all privileges on *.* to 'nodeapp'@'%'";
mysql -u root -e "CREATE DATABASE STUDENTS";
mysql -u root -e "USE STUDENTS; CREATE TABLE students(
    id INT NOT NULL AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    address VARCHAR(255) NOT NULL,
    city VARCHAR(255) NOT NULL,
    state VARCHAR(255) NOT NULL,
    email VARCHAR(255) NOT NULL,
    phone VARCHAR(100) NOT NULL,
    PRIMARY KEY ( id ));"
sed -i 's/.*/bind-address.*bind-address = 0.0.0.0/' /etc/mysql/mysql.conf.d/mysqld.cnf
systemctl enable mysql
service mysql restart
export APP_DB_HOST=$(curl http://169.254.169.254/latest/meta-data/local-ipv4)
export APP_DB_USER=nodeapp
export APP_DB_PASSWORD=student12
export APP_DB_NAME=STUDENTS
export APP_PORT=80
npm start &
echo '#!/bin/bash -xe
cd /home/ubuntu/resources/codebase_partner
export APP_PORT=80
npm start' > /etc/rc.local
chmod +x /etc/rc.local
```

Por último, para instalar la aplicación web y la base de datos necesarias en la máquina virtual, utilicemos el código JavaScript del siguiente enlace: [SolutionCodePOC](#), añadiéndolo en “ Datos de usuario” en la sección de “Detalles avanzados”.

Después de configurar todo, revisamos los detalles de la instancia y procedemos con el lanzamiento.



Con estos pasos, hemos creado y configurado una instancia EC2 en AWS, asegurándonos de que esté correctamente conectada a la red y sea accesible desde Internet.

Una vez que la aplicación web esté instalada y ejecutándose en la instancia EC2, es crucial realizar pruebas para asegurarse de que todo funcione correctamente.

## Tarea 3: probar la implementación

Para asegurarnos de que la aplicación esté funcionando correctamente, abre un navegador web y accede a la dirección IP pública de nuestra instancia EC2. Deberíamos ver la interfaz de la aplicación web, donde podemos realizar operaciones básicas como ver, añadir, eliminar y modificar registros de estudiantes.

A screenshot of a web browser window. The address bar shows 'ec2-3-83-100-27.compute-1.amazonaws.com'. The page title is 'Students'. The main content area features a graduation photo and the text 'XYZ University'. Below it, a 'Welcome' section says 'Use this app to keep track of your student inquiries' and has a 'List of students' link.

A screenshot of a web browser window showing the 'All students' list. The address bar is the same as the previous screenshot. The page title is 'Students'. The main content area shows a table with columns: Name, Address, City, State, Email, and Phone. One row is visible for 'Serhiy' with details: Calle Goya 5, Madrid, Madrid, serhiy@gmail.com, 9009009009. There is an 'edit' button next to the row. At the bottom, there is a green 'Add a new student' button.

All fields are required

Name	<input type="text" value="Serhiy"/>
Name of this student	
Address	<input type="text" value="Calle Goya 5"/>
Address for this student	
City	<input type="text" value="Madrid"/>
City for this student	
State	<input type="text" value="Madrid"/>
State for this student	
Email	<input type="text" value="serhiy@gmail.com"/>
Email for this student	
Phone	<input type="text" value="9009009009"/>
Phone number for this student	
<input type="button" value="Submit"/> <input type="button" value="Delete this student"/>	

## Fase 3: desacoplamiento de los componentes de la aplicación

En esta fase, el objetivo es separar la base de datos y la infraestructura del servidor web para que se ejecuten de forma independiente. Esto mejorará la escalabilidad, seguridad y administración de la aplicación. La aplicación web debe ejecutarse en una máquina virtual independiente, y la base de datos debe ejecutarse en la infraestructura de servicios administrados.

---

### Tarea 1: modificar la configuración de la VPC

La configuración de la VPC, incluyendo la creación de subredes públicas y privadas, y la configuración de la Gateway de Internet y las tablas de enrutamiento, se realizó en la Fase 2. Con esto, ya tenemos las subredes necesarias y la infraestructura de red lista para proceder con el desacoplamiento de los componentes de la aplicación.

---

### Tarea 2: creación y configuración de la base de datos de Amazon RDS

En esta tarea, crearemos una base de datos MySQL utilizando Amazon RDS. Esto permitirá que la base de datos esté gestionada de forma independiente y sea más segura, escalable y fácil de administrar.

#### Notas:

- Permitir que solo la aplicación web acceda a la base de datos.
- No active la supervisión mejorada.

## Referencia

- Fundamentos de la nube de AWS Academy - Laboratorio: cree su servidor de base de datos e interactúe con su base de datos mediante una aplicación

The screenshot shows the AWS RDS 'Create database' wizard. In the first section, 'Elegir un método de creación de base de datos', the 'Creación estándar' option is selected. In the second section, 'Opciones del motor', the 'MySQL' engine is selected. Other engines shown include Aurora (MySQL Compatible), Aurora (PostgreSQL Compatible), MariaDB, PostgreSQL, and Oracle.

Primero, accedemos a la consola de Amazon RDS desde el menú de servicios de AWS. Seleccionamos “Create database” y elegimos el motor de base de datos MySQL. Luego, seleccionamos “Standard Create” para tener más control sobre las opciones de configuración. A continuación, seleccionamos la versión de MySQL que deseé utilizar.

Configuraremos la instancia de la base de datos con los siguientes parámetros:

- Tipo de instancia: db.t3.micro
- Despliegue: Solo en una zona de disponibilidad (Single-AZ)

- Almacenamiento: 20 GB en SSD de uso general (gp2)
- Configuración de la VPC: Seleccionamos la VPC que hemos creado en las tareas anteriores y las subredes privadas para asegurar que la base de datos no sea accesible públicamente.

**▼ Configuración de credenciales**

**Nombre de usuario maestro** [Información](#)  
Escriba un ID de inicio de sesión para el usuario maestro de la instancia de base de datos.

1 a 16 caracteres alfanuméricos. El primer carácter debe ser una letra.

**Administración de credenciales**  
Puede usar AWS Secrets Manager o administrar sus credenciales de usuario maestro.

Administrado en AWS Secrets Manager - más seguro  
RDS genera una contraseña y la administra durante todo su ciclo de vida mediante AWS Secrets Manager.

Autoadministrado  
Cree su propia contraseña o pida a RDS que cree una contraseña para que pueda administrarla.

Generar contraseña automáticamente  
Amazon RDS puede generar una contraseña en su nombre, o bien puede especificar su propia contraseña.

**Contraseña maestra** [Información](#)

Restricciones mínimas: al menos 8 caracteres ASCII imprimibles. No puede contener ninguno de los siguientes símbolos: / ' " @

**Confirmar la contraseña maestra** [Información](#)

Proporcionamos un nombre de usuario “admin” y una contraseña “lab-password” para el administrador de la base de datos. Estos credenciales se usarán más adelante y deben ser almacenados de manera segura.

[EC2](#) > [Grupos de seguridad](#) > sg-00df9adb9afbb1dd6

**sg-00df9adb9afbb1dd6 - db-sg**

Detalles			
Nombre del grupo de seguridad <a href="#">db-sg</a>	ID del grupo de seguridad <a href="#">sg-00df9adb9afbb1dd6</a>	Descripción <a href="#">Permite acceso a base de datos</a>	ID de la VPC <a href="#">vpc-05cc22f6c0fa292fd</a>
Propietario <a href="#">951516663729</a>	Número de reglas de entrada 1 Entrada de permiso	Número de reglas de salida 1 Entrada de permiso	

[Reglas de entrada](#) [Reglas de salida](#) [Etiquetas](#)

Reglas de entrada (1)										
<a href="#">G</a> Administrar etiquetas <a href="#">Editar</a>										
<input type="text"/> Buscar										
<input type="checkbox"/>	Name	ID de la regla del grupo	Versión de IP	Tipo	Protocolo	Intervalo de puertos	Origen	Descripción		
<input type="checkbox"/>	-	sgr-06b1b7da754ebbeb8	-	MySQL/Aurora	TCP	3306	sg-053d95874181d9a...			

A continuación, configuraremos los grupos de seguridad para la instancia RDS. Creamos un nuevo grupo de seguridad o selecciona uno existente que permita el tráfico entrante en el puerto 3306 (puerto predeterminado de MySQL) desde el grupo de seguridad asignado a las instancias EC2 que

alojan la aplicación web. Esto asegurará que solo las instancias autorizadas puedan comunicarse con la base de datos.

Después de configurar todos los parámetros necesarios, selecciona “Create database” para lanzar la instancia RDS. AWS comenzará a aprovisionar la base de datos, lo cual puede tomar unos minutos.

Conectividad y seguridad		
<b>Punto de enlace y puerto</b>	<b>Redes</b>	<b>Seguridad</b>
Punto de enlace database-student.cnwxtfzseeg1.us-east-1.rds.amazonaws.com	Zona de disponibilidad us-east-1a	Grupos de seguridad de la VPC <b>db-sg (sg-00df9adb9afbb1dd6)</b> Activo
Puerto 3306	VPC proyecto-vpc (vpc-05cc22f6c0fa292fd)	rds-ec2-1 (sg-0fab877b63822219) Activo
	Grupo de subredes db-subnet-group	Accesible públicamente No
	Subredes subnet-08396fbdd18f63116 subnet-0d1c9c8d2e62e1f24	Entidad de certificación <a href="#">Información</a> rds-ca-rsa2048-g1
	Tipo de red IPv4	Fecha de la entidad de certificación May 26, 2061, 01:34 (UTC+02:00)
		Fecha de expiración del certificado de instancia de base de datos June 23, 2025, 15:07 (UTC+02:00)

Una vez que la base de datos esté disponible, anotamos el endpoint de la base de datos, ya que lo necesitaremos para configurar la conexión desde la aplicación web. Este endpoint es la dirección que la aplicación utilizará para conectarse a la base de datos RDS.

Con la base de datos de Amazon RDS configurada y en funcionamiento, hemos completado esta tarea, preparando el entorno para la integración segura y eficiente de la aplicación web con la base de datos.

## Tarea 3: configuración del entorno de desarrollo

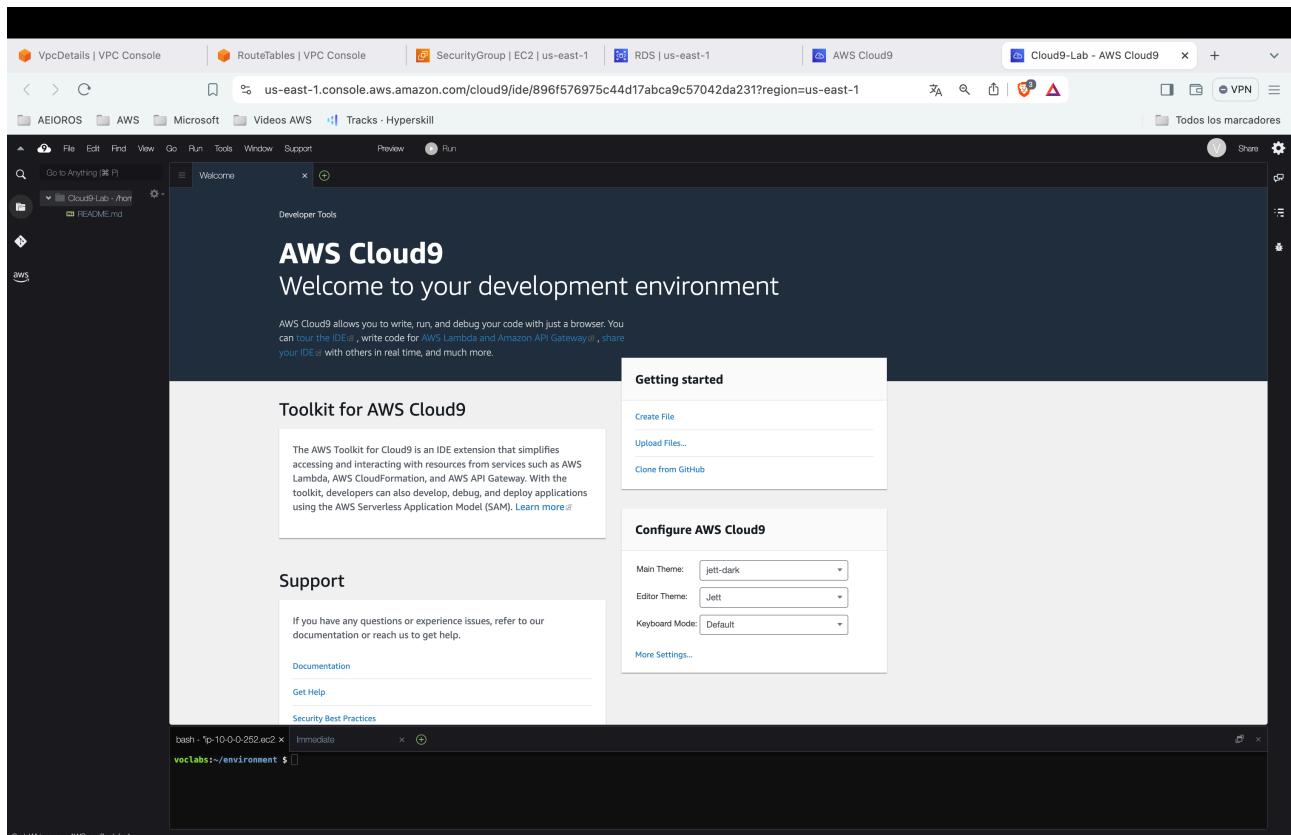
En esta tarea, aprovisionaremos un entorno de desarrollo utilizando AWS Cloud9. Este entorno facilitará la administración y ejecución de comandos AWS CLI, lo que será útil para las tareas de configuración y gestión de los servicios de AWS.

## Notas:

- Utilice una instancia t3.micro para el entorno AWS Cloud9.
- Utilice Secure Shell (SSH) para conectarse al entorno.

## Referencia

- [Crear el entorno de Cloud9](#)



Primero, accedemos a la consola de AWS Cloud9 desde el menú de servicios de AWS. Selecciona “Create environment” para comenzar con la configuración de un nuevo entorno de desarrollo. Proporcionamos un nombre para nuestro entorno, “Cloud9-Lab”.

En la siguiente pantalla, seleccionamos la opción para crear un nuevo entorno con una instancia EC2. Elimos una instancia de tipo t3.micro para mantener los costos bajos y aprovechar el nivel gratuito de AWS. Configuraremos la instancia para que esté dentro de la misma VPC y subredes que configuro anteriormente. Esto asegurará que el entorno de desarrollo tenga acceso adecuado a los recursos de la VPC, incluida la base de datos RDS.

Asegúrate de que la instancia de Cloud9 tenga una IP pública asignada para que puedas acceder al entorno desde cualquier lugar. En la sección de configuración de red, selecciona la VPC correcta y una de las subredes públicas.

The screenshot shows the AWS Cloud9 interface. On the left, there's a sidebar with 'Mis entornos', 'Compartido conmigo', 'Todos los entornos de la cuenta', and 'Documentación'. The main area has a blue header bar with the text 'CreandoCloud9-Lab. Esto puede demorar varios minutos. Mientras espera, consulte Prácticas recomendadas para usar AWS Cloud9'. Below it, a table titled 'Entornos (1)' lists one environment: 'Cloud9-Lab' (Abrir), which is an 'Instancia de EC2' with 'Secure Shell (SSH)' connection type and 'Propietario' permission, ARN: arn:aws:sts::951516663729:assumed-role/voclabs/user3082628=Serhiy\_Holovasin.

Una vez configurado el entorno, selecciona “Create environment”. AWS Cloud9 comenzará a aprovisionar la instancia y configurará el entorno de desarrollo. Este proceso puede tomar unos minutos.

The screenshot shows the AWS Cloud9 IDE interface. It features a terminal window with the following text:

```
1  _____
2 / \ \ \ \ \ \
3 / \ \ \ \ \ \
4 / \ \ \ \ \ \
5 / \ \ \ \ \ \
6 / \ \ \ \ \ \
7 / \ \ \ \ \ \
8 / \ \ \ \ \ \
9 Hi there! Welcome to AWS Cloud9!
10 To get started, create some files, play with the terminal,
11 or visit https://docs.aws.amazon.com/console/cloud9/ for our documentation.
12
13 Happy coding!
14
15
```

Below the terminal, there's a preview pane showing a file named 'README.md' with some text and a small diagram. The bottom right corner of the interface shows icons for 'aws', 'aws - ip-10-0-0-252.ec2.i', and a '+' sign.

Cuando el entorno esté listo, Cloud9 abrirá un IDE basado en navegador donde podremos escribir, ejecutar y depurar código. Además, el entorno viene preconfigurado con AWS CLI, lo que facilita la ejecución de comandos para gestionar nuestros servicios de AWS.

Dentro del entorno de Cloud9, verificamos la configuración inicial y aseguramos de que tenemos acceso a los servicios de AWS necesarios. Puedes hacer esto ejecutando algunos comandos básicos de AWS CLI.

The screenshot shows a terminal window in AWS Cloud9. The user is in their home directory (~). They run several commands:

```
voclabs:~/environment $ pwd
/home/ec2-user/environment
voclabs:~/environment $ ls-la
bash: ls-la: command not found
voclabs:~/environment $ ls -la
total 24
drwxr-xr-x. 3 ec2-user ec2-user 50 Jun 23 16:29 .
drwx----- 13 ec2-user ec2-user 16384 Jun 26 17:54 ..
drwxr-xr-x. 5 ec2-user ec2-user 184 Jun 23 15:58 .c9
-rw-r--r--. 1 ec2-user ec2-user 569 Jan 1 2000 README.md
-rw-r--r--. 1 ec2-user ec2-user 2402 Jun 23 16:33 data.sql
voclabs:~/environment $
```

Con el entorno de desarrollo de AWS Cloud9 configurado, tendremos una plataforma robusta y accesible para administrar y desarrollar nuestra aplicación web.

## Tarea 4: aprovisionamiento del Secrets Manager

En esta tarea, utilizaremos AWS Secrets Manager para gestionar de forma segura las credenciales de la base de datos MySQL mediante un script dentro del entorno de AWS Cloud9.

Utilicemos *Script-1* del enlace siguiente para crear un secreto en Secrets Manager mediante AWS CLI: [AWS Cloud9 Scripts](#).

**Nota:** Este archivo .yml también contiene los scripts que utilizaremos en tareas posteriores.

### Referencia

- [create-secret en la referencia de comandos de AWS CLI para AWS Secrets Manager](#)

Primero, accedemos a nuestro entorno de AWS Cloud9 que configuremos en la tarea anterior. Una vez dentro del entorno, abrimos una nueva terminal para ejecutar el script que creará el secreto en AWS Secrets Manager.

A continuación, copiamos y pegamos el siguiente script en la terminal de AWS Cloud9. Asegúrándole de reemplazar los valores de los marcadores de posición con los valores reales que hemos utilizado para configurar nuestra base de datos RDS (nombre de usuario, contraseña, endpoint y nombre de la base de datos):

```
# Script para crear un secreto en AWS Secrets Manager
aws secretsmanager create-secret \
--name Mydbsecret \
--description "Database secret for web app" \
--secret-string "{\"user\":\"admin\",\"password\":\"lab-password\",\"host\":\"database-student.cnwxtfzseeg1.us-east-1.rds.amazonaws.com\",\"db\":\"STUDENTS\"}"
```

Este script crea un nuevo secreto llamado “Mydbsecret” en AWS Secrets Manager, con las credenciales necesarias para acceder a tu base de datos MySQL.

```
aws -v ip-10-0-0-252.ec2.i x +  
voclabs:~/environment $ aws secretsmanager create-secret \  
>   --name Mydbsecret \  
>   --description "Database secret for web app" \  
>   --secret-string "{\"user\":\"admin\",\"password\":\"lab-password\",\"host\":\"database-student.cnwxtfzseeg1.us-east-1.rds.amazonaws.com\",\"db\":\"STUDENTS\"}"  
{  
    "ARN": "arn:aws:secretsmanager:us-east-1:951516663729:secret:Mydbsecret-AhoGdm",  
    "Name": "Mydbsecret",  
    "VersionId": "8d042b91-d8fc-479a-949e-e7d0fb3563d5"  
}  
voclabs:~/environment $
```



Después de ejecutar el script, el secreto será creado y almacenado de forma segura en AWS Secrets Manager. El secreto que se crea en este paso almacena las credenciales de la base de datos, que la aplicación web utilizará a través de un rol de AWS Identity and Access Management (IAM) llamado LabRole. Esto mejora la seguridad al no almacenar credenciales en la aplicación o en la base de datos. LabRole se creó previamente en el entorno de laboratorio. El rol facilita las interacciones seguras entre los servicios de AWS. El rol ya incluye las políticas de permisos apropiadas.

Ahora, necesitamos configurar nuestra aplicación web para que utilice este secreto.

---

## Tarea 5: aprovisionamiento de una nueva instancia para el servidor web

En esta tarea, crearemos una nueva instancia EC2 que alojará la aplicación web. Esta instancia se configurará para utilizar las credenciales de la base de datos gestionadas por AWS Secrets Manager.

Para instalar la aplicación web requerida en la máquina virtual, utilice el código JavaScript del siguiente enlace: [Solution Code for the App Server](#)

```
#!/bin/bash -xe
apt update -y
apt install nodejs unzip wget npm mysql-client -y
# wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACCAP1-1-DEV/
code.zip -P /home/ubuntu
wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-
ACCAP1-1-79581/1-lab-capstone-project-1/code.zip -P /home/ubuntu
cd /home/ubuntu
unzip code.zip -x "resources/codebase_partner/node_modules/*"
cd resources/codebase_partner
npm install aws aws-sdk
export APP_PORT=80
npm start &
echo '#!/bin/bash -xe
cd /home/ubuntu/resources/codebase_partner
export APP_PORT=80
npm start' > /etc/rc.local
chmod +x /etc/rc.local
```

Para el perfil de AWS Identity and Access Management (IAM) en la instancia de EC2, adjunte el perfil *LabInstanceProfile* existente. Este perfil adjunta un rol de IAM llamado *LabRole* a la instancia para que pueda obtener el secreto de forma segura.

**Nota:** Opcionalmente, puede seguir utilizando la máquina virtual existente para la aplicación web. Sin embargo, tendrá que volver a configurar la aplicación para conectarse a Amazon RDS.

### Referencia

- Fundamentos de la nube de AWS Academy - Laboratorio: cree su servidor de base de datos e interactúe con su base de datos mediante una aplicación

En esta tarea, repetiremos los pasos de la Tarea 2 de la Fase 2 para aprovisionar una nueva instancia EC2, pero esta vez utilizaremos un script diferente para configurar la instancia y desplegar la aplicación web. También en la sección de configuración de IAM, asignamos el rol LabInstanceProfile a la instancia EC2. Este perfil de IAM le permitirá a la instancia obtener los secretos almacenados en AWS Secrets Manager de forma segura.

Después de configurar todos los parámetros necesarios, revisamos y lanzamos la instancia.

The screenshot shows the AWS EC2 Instances page with the following details:

Name	ID de la instancia	Estado de la i...	Tipo de inst...	Comprobación de	Estado de la al...	Zona de dispon...	DNS de IPv4 pública
WebServer1	i-08eafdf87885988c8	En ejecución	t2.micro	2/2 comprobador	Ver alarmas	us-east-1b	ec2-44-203-70-181.co...
Webserver_sinBD	i-095e164d9f7841363	En ejecución	t2.micro	-	Ver alarmas	us-east-1a	ec2-34-200-226-251.co...
aws-cloud9-Cloud9-Lab...	i-0d1dac276b3ff9892	En ejecución	t3.small	2/2 comprobador	Ver alarmas	us-east-1a	ec2-3-237-67-121.com...

## Tarea 6: migrar la base de datos

Para migrar los datos de la base de datos original que se encuentra en la instancia EC2 a la nueva base de datos de Amazon RDS, utilizaremos los comandos proporcionados en un entorno AWS Cloud9. Este enfoque asegura que todos los datos existentes se transfieran de manera segura y eficiente a la nueva infraestructura gestionada.

Utilizaremos Script-3 del archivo de scripts de AWS Cloud9 (cloud9-scripts.yml) para migrar los datos originales a la base de datos de Amazon RDS. Recuerde que anteriormente utilizó un script de este archivo para crear el secreto en Secrets Manager.

```
Script -3
## Migration
# Following command exports the data from existing server.
# Prerequisites - This script expects mysql-client, mysqlfdump to be present
# AWS Cloud9 environment already has all the prerequisites installed for running these scripts

# Replace the <EC2instancePrivateip> with internal IP address of the EC2 instance
#(CapstonePOC) created in Phase-2 earlier.
# Provide the password when prompted
```

```
mysqldump -h 10.0.2.86 -u nodeapp -p --databases STUDENTS > data.sql
```

```
#Following command imports the data into RDS database. Replace <RDSEndpoint> with the
RDS Database endpoint you noted after RDS Database created in earlier steps.
#when prompted, enter password you provided during the time of database creation
mysql -h database-student.cnwxtfzseeg1.us-east-1.rds.amazonaws.com -u admin -p STUDENTS
< data.sql
```

## Referencia

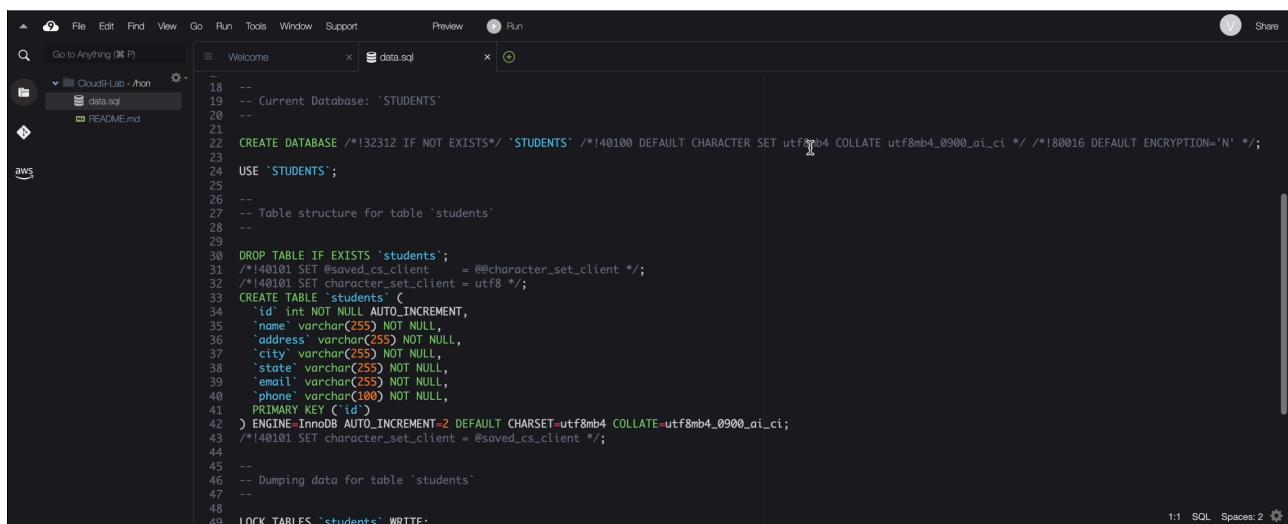
- Arquitectura en la nube de AWS Academy - Laboratorio: migración de una base de datos a Amazon RDS

Primero, accedemos a nuestro entorno de AWS Cloud9, asegurando de que tenemos acceso tanto a la instancia EC2 original como a la nueva instancia de Amazon RDS desde este entorno.

En la terminal de AWS Cloud9, ejecutamos el siguiente script para exportar los datos de la base de datos existente en la instancia EC2. Reemplaza <EC2instancePrivateip> con la dirección IP privada de la instancia EC2 donde se encuentra la base de datos MySQL original.

```
vocabls:~/environment $ mysqldump -h 10.0.2.86 -u nodeapp -p --databases STUDENTS > data.sql
Enter password:
```

Proporcionamos la contraseña de la base de datos cuando se nos solicite. Este comando creará un archivo llamado data.sql que contiene todos los datos y estructuras de la base de datos.



```
18 --
19 -- Current Database: `STUDENTS`
20 --
21
22 CREATE DATABASE /*!32312 IF NOT EXISTS*/ `STUDENTS` /*!40100 DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci */ /*!180016 DEFAULT ENCRYPTION='N' */;
23
24 USE `STUDENTS`;
25
26 --
27 -- Table structure for table `students`.
28 --
29
30 DROP TABLE IF EXISTS `students`;
31 /*!40101 SET @saved_cs_client      = @@character_set_client */;
32 /*!40101 SET character_set_client = utf8 */;
33 CREATE TABLE `students` (
34   `id` int NOT NULL AUTO_INCREMENT,
35   `name` varchar(255) NOT NULL,
36   `address` varchar(255) NOT NULL,
37   `city` varchar(255) NOT NULL,
38   `state` varchar(255) NOT NULL,
39   `email` varchar(255) NOT NULL,
40   `phone` varchar(100) NOT NULL,
41   PRIMARY KEY (`id`)
42 ) ENGINE=InnoDB AUTO_INCREMENT=2 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
43 /*!40101 SET character_set_client = @saved_cs_client */;
44
45 --
46 -- Dumping data for table `students`
47 --
48 LOCK TABLES `students` WRITE;
```

A continuación, ejecutamos el siguiente comando para importar los datos a la nueva base de datos Amazon RDS, reemplazando <RDSEndpoint> con el endpoint de la base de datos RDS que hemos anotado al crear la instancia de RDS.

```
vocabls:~/environment $ mysql -h database-student.cnwxtfzseeg1.us-east-1.rds.amazonaws.com -u admin -p STUDENTS < data.sql
Enter password:
```

Proporcionamos la contraseña de la base de datos RDS cuando se nos solicite. Este comando cargará el volcado de la base de datos en la nueva instancia de RDS.

Una vez completada la importación, verificamos que todos los datos se hayan transferido correctamente. Podemos hacer esto ejecutando algunas consultas de verificación en la nueva base de datos RDS desde nuestro entorno AWS Cloud9.

The screenshot shows a Cloud9 IDE interface with two terminal tabs: 'data.sql' and 'README.md'. The 'data.sql' tab contains MySQL commands to connect to an RDS instance and show databases, resulting in a list of databases including 'STUDENTS', 'information\_schema', 'mysql', 'performance\_schema', and 'sys'. The 'README.md' tab contains a brief description of the task.

The browser window displays the 'XYZ University' student management application. It has a dark theme with a header in English ('Home', 'Students list'). The main page shows a welcome message and a table titled 'List of students' with one row of data (Serhiy, Calle Goya 5, Madrid, serhiy@gmail.com, 9009009009). Below the table is a button labeled 'Add a new student'.

Con esta tarea completada, hemos migrado exitosamente los datos de la base de datos original a la nueva base de datos Amazon RDS.

## Tarea 7: probar la aplicación

Con la migración de la base de datos completada, el siguiente paso es probar la aplicación para asegurarnos de que todo esté funcionando correctamente con la nueva configuración de la base de datos. Este paso es importante para verificar que la aplicación puede conectarse a la nueva instancia de Amazon RDS y realizar todas las operaciones necesarias sin problemas.

Pero antes de probar la aplicación, necesitamos asegurarnos de que la instancia EC2 puede conectarse correctamente a la nueva base de datos Amazon RDS. Este paso implica la configuración de los grupos de seguridad y la actualización de la configuración de la aplicación web para que apunte a la nueva base de datos.

AWS | Servicios | Buscar [Opción+S] | ☰ ⓘ ⓘ ⓘ ⓘ

RDS > Bases de datos > Configurar conexión de EC2

Paso 1: Configurar conexión de EC2

Paso 2: Revisar y confirmar

## Configurar conexión de EC2 Información

### Seleccionar instancia de EC2

Base de datos: database-student

Instancia de EC2:

Elija la instancia de EC2 para conectarse a esta base de datos. Solo se muestran las instancias de EC2 de la misma VPC que la base de datos. Si no hay ninguna instancia de EC2 disponible en la misma VPC, puede crear una nueva instancia de EC2.

i-095e164d9d7841363  
Webserver\_sinBD us-east-1

Crear instancia EC2

**Continuar**

AWS | Servicios | Buscar [Opción+S] | ☰ ⓘ ⓘ ⓘ ⓘ

Paso 1: Configurar conexión de EC2

Paso 2: Revisar y confirmar

## Revisar y confirmar

### Resumen de conexión Información

Está configurando una conexión entre la base de datos de RDS database-student y la instancia de EC2 i-095e164d9d7841363.

Para configurar una conexión entre la base de datos y la instancia de EC2, el grupo de seguridad de VPC *rds-ec2-1* se agrega a la base de datos y el grupo de seguridad de VPC *ec2-rds-1* se agrega a la instancia de EC2.

VPC: vpc-05cc22f6c0fa292fd (proyecto-vpc)

Grupo de seguridad: rds-ec2-1 (regla de conexión)

RDS

database-student Puerto: 3306

Grupo de seguridad: ec2-rds-1 (regla de conexión)

EC2

i-095e164d9d7841363

En negrita, se indica que se está realizando una adición para configurar una conexión.

### Cambios en la base de datos de RDS: database-student

Atributo	Valor actual	Valor nuevo
Grupo de seguridad	db-sg	db-sg, rds-ec2-1

### Cambios en la instancia de EC2: i-095e164d9d7841363

Atributo	Valor actual	Valor nuevo
Grupo de seguridad	webserver-sg	webserver-sg, ec2-rds-1

**Configurar**

The screenshot shows the AWS RDS console with the following details:

- Configuración de la conexión establecida con éxito para la base de datos RDS database-student y Instancia de EC2 I-095e164d9d7841363**
- Presentamos Aurora optimizado para las operaciones de E/S**: Aurora optimizado para las operaciones de E/S es una nueva configuración de almacenamiento en clúster que ofrece precios predecibles para todas las aplicaciones y una mejor relación calidad-precio, con un ahorro de hasta el 40 % en costos para aplicaciones que requieren un uso intensivo de las operaciones de E/S.
- Bases de datos (1)**: database-student (Available, MySQL Community, us-east-1a, db.t3.micro, 2.94%, 2 Conexiones)

Solo después de estos pasos podemos comprobar correcto funcionamiento de nuestra app:

Vemos que todas operaciones se realizan sin problemas. Eso significa que la aplicación web está funcionando correctamente con la nueva configuración de la base de datos en Amazon RDS. Este paso asegura que la migración de datos fue exitosa y que la aplicación es capaz de interactuar con la base de datos de manera eficiente.

## Fase 4: implementar la alta disponibilidad y escalabilidad

En esta fase, completaremos el diseño y cumpliremos con los requisitos restantes de la solución. El objetivo es utilizar los componentes clave creados en fases anteriores para construir una arquitectura escalable y de alta disponibilidad.

**Consejo:** Utilice un mínimo de dos zonas de disponibilidad.

The screenshot shows a web browser window with the following details:

- Address bar:** ec2-34-200-226-251.compute-1.amazonaws.com/students
- Header:** XYZ University (Home, Students list)
- Content:** All students table with two rows:
 

Name	Address	City	State	Email	Phone	Action
Serhiy	Calle Goya 5	Madrid	Madrid	serhiy@gmail.com	9009009009	<b>edit</b>
Felipe	Calle Sol 45	Madrid	Madrid	felipe@gmail.com	1112223334	<b>edit</b>

## Referencia

- Arquitectura en la nube de AWS Academy - Laboratorio: creación de un entorno de alta disponibilidad

## Tarea 1: crear un Application Load Balancer

El primer paso para asegurar la alta disponibilidad y escalabilidad de la aplicación web es implementar un Application Load Balancer (ALB). Este componente distribuirá el tráfico entre múltiples instancias EC2, garantizando que la carga de trabajo esté equilibrada y que la aplicación permanezca accesible incluso si una instancia falla.

Para comenzar, necesitamos configurar un grupo de seguridad que permita el tráfico HTTP entrante al ALB. Accedemos a la consola de EC2 y seleccionamos “Security Groups” en el menú de navegación. Pulsamos en “Create security group” y proporcionamos un nombre y una descripción para el grupo de seguridad, por ejemplo “alb-sg”. Asegúramos de seleccionar la VPC correcta. Luego, añadimos una regla de entrada que permita el tráfico HTTP (puerto 80) desde cualquier origen (0.0.0.0/0). Esta configuración permitirá que el ALB reciba tráfico desde cualquier usuario de Internet. Una vez configurado, hacemos clic en “Create security group” para finalizar.

The screenshot shows the AWS Security Groups configuration page for a group named 'sg-0a2892e87bb40111f - alb-sg'. The 'Detalles' tab is selected, displaying information such as the group name ('alb-sg'), ID ('sg-0a2892e87bb40111f'), owner ('951516663729'), and a single inbound rule allowing HTTP traffic on port 80 from anywhere. Below this, the 'Reglas de entrada (1)' section lists the rule: Name 'sgr-0f605197cfaf10d7', Version of IP 'IPv4', Type 'HTTP', Protocol 'TCP', Port '80', Origin '0.0.0.0/0', and Description 'Permite solo el trafico http'. There are tabs for 'Reglas de salida' and 'Etiquetas' at the bottom.

A continuación, creamos un grupo de destino que el ALB utilizará para dirigir el tráfico a las instancias EC2. En la consola de EC2, seleccionamos “Target Groups” y hacemos clic en “Create target group”. Seleccionamos “Instances” como tipo de destino y proporcionamos un nombre para el grupo de destino, por ejemplo “ec2-tg”. Configuraremos el puerto 80 como el puerto de destino y selecciona el protocolo HTTP. Asegúramos de seleccionar la VPC correcta. Hacemos clic en “Next” y en la siguiente pantalla, seleccionamos la instancia EC2 que ejecuta nuestra aplicación web. Registraremos esta instancia con el grupo de destino para que el ALB pueda dirigir el tráfico hacia ella.

Ahora accedemos a la consola de EC2 y seleccionamos “Load Balancers” en el menú de navegación. Hacemos clic en “Create Load Balancer” y seleccionamos “Application Load Balancer” entre las opciones disponibles. Proporcionamos un nombre descriptivo para el ALB y selecciona “Internet-facing” como tipo de esquema, lo que permitirá que el ALB sea accesible desde Internet. A continuación, configuraremos las zonas de disponibilidad seleccionando al menos dos subredes públicas de la VPC que hemos configurado anteriormente. Esta configuración asegura que el ALB pueda distribuir el tráfico entre múltiples zonas de disponibilidad, mejorando la resiliencia y disponibilidad de la aplicación.

En la siguiente etapa, configuraremos un listener en el puerto 80 para manejar el tráfico HTTP entrante. Un listener es una entidad que comprueba las conexiones de los clientes utilizando el puerto y el protocolo que configures.

Luego, seleccionamos el grupo de destino (target group) Finalmente, seleccionamos un grupo de seguridad creado anteriormente para nuestro ALB.

Esto asegura que el ALB pueda recibir tráfico de cualquier usuario de Internet. Revisamos todas las configuraciones y, si todo está correcto, hacemos clic en “Create” para lanzar el Application Load Balancer.

Con el ALB en funcionamiento, hemos mejorado significativamente la disponibilidad y la capacidad de nuestra aplicación web para manejar grandes volúmenes de tráfico. En la siguiente tarea nos centraremos en la implementación de Amazon EC2 Auto Scaling para asegurar que la aplicación pueda escalar automáticamente según la demanda.

## Tarea 2: implementación de Amazon EC2 Auto Scaling

El siguiente paso es configurar Amazon EC2 Auto Scaling para asegurar que nuestra aplicación pueda manejar aumentos en la carga de trabajo escalando automáticamente el número de instancias EC2 según la demanda..

### Consejos:

- Utilice una política de seguimiento de destino.
- Ajuste el tamaño del grupo de Auto Scaling según sus necesidades estimadas.
- Puede utilizar inicialmente los valores predeterminados (p. ej., para el tamaño de grupo y la utilización de la CPU) y ajustarlos posteriormente según sea necesario.

### Referencia

- Arquitectura en la nube de AWS Academy - Laboratorio: creación de un entorno de alta disponibilidad

Primero, crearemos una plantilla de lanzamiento que especificará cómo deben configurarse las nuevas instancias EC2. Accedemos a la consola de EC2 y seleccionamos la instancia con nuestra app, en el menú de “Acciones” elegimos “Imagen y Plantillas”. Hacemos clic en “Crear plantilla a partir de una instancia” y proporcionamos un nombre y una descripción para la plantilla.

Seleccionamos la misma AMI para las instancias actuales y configuramos el tipo de instancia, por ejemplo, t2.micro. En la sección de configuración de red, seleccionamos la VPC y las subredes públicas adecuadas. Asegúramos de especificar el grupo de seguridad correcto que permite el tráfico HTTP y SSH.

Guardamos la plantilla de lanzamiento una vez que hayas completado todas las configuraciones.

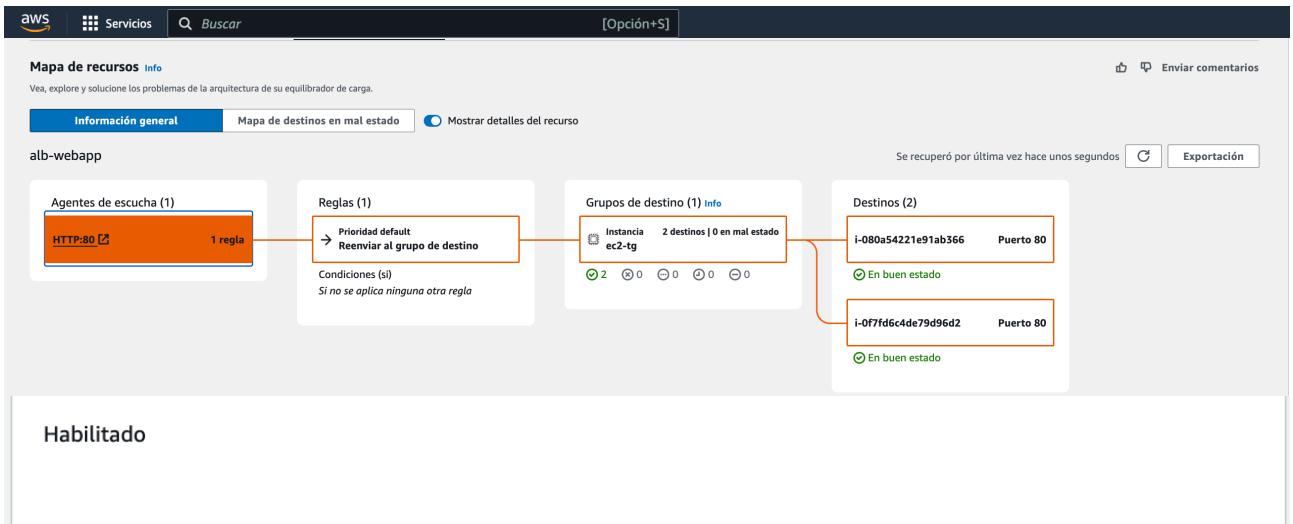
A continuación, creamos un grupo de Auto Scaling utilizando la plantilla de lanzamiento. En la consola de EC2, seleccionamos “Auto Scaling Groups” y pulsamos en “Create Auto Scaling group”. Proporcionamos un nombre para el grupo de Auto Scaling y seleccionamos la plantilla de lanzamiento que acabamos de crear. Especificamos la VPC y seleccionamos al menos dos subredes públicas para distribuir las instancias a través de múltiples zonas de disponibilidad.

Configuramos las opciones de escalado. Seleccionamos una política de escalado basada en el seguimiento del destino, por ejemplo, utiliza la métrica de utilización de la CPU. Configuramos el objetivo de utilización de la CPU al 50% para empezar. Esto significa que el grupo de Auto Scaling lanzará o terminará instancias automáticamente para mantener la utilización de la CPU en un nivel óptimo.

Definimos el tamaño del grupo de Auto Scaling. Específicamos el número mínimo de instancias (por ejemplo, 2), el número máximo de instancias (por ejemplo, 6) y el tamaño deseado del grupo (por ejemplo, 2). Estas configuraciones asegurarán que siempre haya al menos dos instancias ejecutándose, pero permitirán que el grupo escale hasta seis instancias si la demanda aumenta.

Revisamos las configuraciones y pulsamos en “Create Auto Scaling group”. El grupo de Auto Scaling comenzará a lanzar instancias según la configuración especificada y el tráfico que reciba.

Con el Auto Scaling configurado, hemos mejorado la capacidad de nuestra aplicación web para manejar grandes volúmenes de tráfico al escalar automáticamente el número de instancias EC2.



## Habilitado

This screenshot shows the 'Paso 6 - opcional' (Optional Step 6) section of the Auto Scaling configuration wizard. It includes fields for 'VPC' (selected VPC: 'vpc-05cc22f6c0fa292fd (proyecto-vpc) 10.0.0.0/16'), 'Zonas de disponibilidad y subredes' (Availability zones and subnets: 'us-east-1a | subnet-08d923321893fb324 (proyecto-subnet-public1-us-east-1a) 10.0.0.0/24' and 'us-east-1b | subnet-02b3f047da3518296 (proyecto-subnet-public2-us-east-1b) 10.0.2.0/24'), and a 'Crear una subred' (Create a subnet) button. At the bottom are buttons for 'Cancelar' (Cancel), 'Omitir para revisar' (Skip for review), 'Anterior' (Previous), and a highlighted 'Siguiente' (Next) button.

This screenshot shows the 'Paso 7' (Step 7) section of the Auto Scaling configuration wizard. It includes fields for 'Elegir las opciones de lanzamiento de instancias' (Select instance launch options) and 'Configurar las opciones avanzadas' (Configure advanced options). The 'Configurar las opciones avanzadas' section is expanded, showing options for 'Balance de carga' (Load balancing) and 'Asociar a un balanceador de carga existente' (Associate with an existing load balancer). Under 'Balance de carga', there are three options: 'No se encontró ningún balanceador de carga' (No load balancer found), 'Asociar a un balanceador de carga existente' (Associate with an existing load balancer), and 'Asociar a un nuevo balanceador de carga' (Associate with a new load balancer). The 'Asociar a un balanceador de carga existente' option is selected. Under 'Asociar a un balanceador de carga existente', there are two options: 'Elegir entre los grupos de destino del balanceador de carga' (Select from the load balancer's destination groups) and 'Elegir entre balanceadores de carga clásicos' (Select from classic load balancers). The first option is selected. At the bottom are buttons for 'Cancelar' (Cancel), 'Omitir para revisar' (Skip for review), 'Anterior' (Previous), and a highlighted 'Siguiente' (Next) button.

The screenshot shows a web application for managing student information. At the top, there's a header with the XYZ University logo and a "Students list" link. Below the header, a blue banner displays the message "All fields are required". The main form consists of several input fields:

- Name:** Lara
- Address:** Florida 6
- City:** El Escorial
- State:** Madrid
- Email:** lara@gmail.com
- Phone:** 8907654321

Below the form is a summary table with the following details:

Tipo de equilibrador de carga Aplicación	Estado Activó	VPC <a href="#">vpc-05cc22f6c0fa292fd</a>	Tipo de dirección IP IPv4
Esquema Internet-facing	Zona hospedada Z35SXDOTRQ7X7K	Zonas de disponibilidad <a href="#">subnet-02b3f047da3518296</a> us-east-1b (use1-az2) <a href="#">subnet-08d923321895fb324</a> us-east-1a (use1-az1)	Fecha creada 23 de junio de 2024, 19:34 (UTC+02:00)
ARN del equilibrador de carga	Nombre de DNS Info <a href="#">alb-webapp-1028279269.us-east-1.elb.amazonaws.com</a> (Registro A)		
<a href="#">arn:aws:elasticloadbalancing:us-east-1:951516663729:loadbalancer/app/alb-webapp/af2832c74168bc96</a>			

A "Submit" button is located at the bottom left of the form area.

según la demanda. La próxima tarea se centrará en acceder y probar la aplicación para asegurar que todo funcione correctamente con la configuración de alta disponibilidad y escalabilidad.

The screenshot shows a browser window with the XYZ University Students list page. The URL in the address bar is [alb-webapp-1028279269.us-east-1.elb.amazonaws.com/students](http://alb-webapp-1028279269.us-east-1.elb.amazonaws.com/students). The page title is "Students". The main content area displays the "All students" section with a single entry:

Name	Address	City	State	Email	Phone	Action
Serhiy	Calle Goya 5	Madrid	Madrid	serhiy@gmail.com	9009009009	<a href="#">edit</a>

At the bottom of the table, there's a green button labeled "Add a new student". The browser's toolbar and address bar are visible at the top.

## Tarea 3: acceder a la aplicación

Después de configurar el Application Load Balancer (ALB) y el Auto Scaling, el siguiente paso es acceder a la aplicación y realizar pruebas para asegurar que todo funcione correctamente.

Primero, obténemos la URL del Application Load Balancer. Accedemos a la consola de EC2, seleccionamos “Load Balancers” en el menú de navegación y encontramos el ALB que creaste anteriormente. La URL del ALB estará en la columna “DNS Name”. Copia esta URL.

Abrimos un navegador web y pegamos la URL del ALB en la barra de direcciones. Esta URL nos permitirá acceder a la aplicación que ahora está balanceada y escalada automáticamente.

## Tarea 4: prueba de carga de la aplicación

Para realizar una prueba de carga en la aplicación y supervisar cómo responde al aumento del tráfico, utilizaremos el paquete loadtest. Este paquete nos permitirá simular múltiples usuarios accediendo a la aplicación al mismo tiempo, y verificar que el escalado automático funciona correctamente.

Utilicemos *Script-2* del archivo AWS Cloud9 Scripts (cloud9-scripts.yml) para realizar la prueba de carga. Recuerde que utilizó scripts de este archivo en tareas anteriores.

### Notas:

- Acceda a la aplicación web desde el navegador utilizando la URL del equilibrador de carga.
- Utilice AWS Cloud9 para ejecutar los scripts de pruebas de carga contra el equilibrador de carga.

The screenshot shows a web application interface for 'XYZ University'. At the top, there is a dark header with the university's name and navigation links for 'Home' and 'Students list'. Below the header, a banner features a photo of graduates throwing caps in the air. The main content area is titled 'All students' and displays a table with two rows of student data. Each row includes columns for Name, Address, City, State, Email, and Phone, along with an 'edit' button. At the bottom left of the table is a green 'Add a new student' button.

Name	Address	City	State	Email	Phone
Serhiy	Calle Goya 5	Madrid	Madrid	serhiy@gmail.com	9009009009
Lara	Florida 6	El Escorial	Madrid	lara@gmail.com	8907654321

**Add a new student**

## Referencia

- [Repositorio de herramientas de prueba de carga en GitHub](#)

Script-2:

```
## Load testing
```

```
#Following command installs #loadtest package to perform load testing on the application
#Prerequisites are mentioned in the resources section
```

```
npm install -g loadtest
```

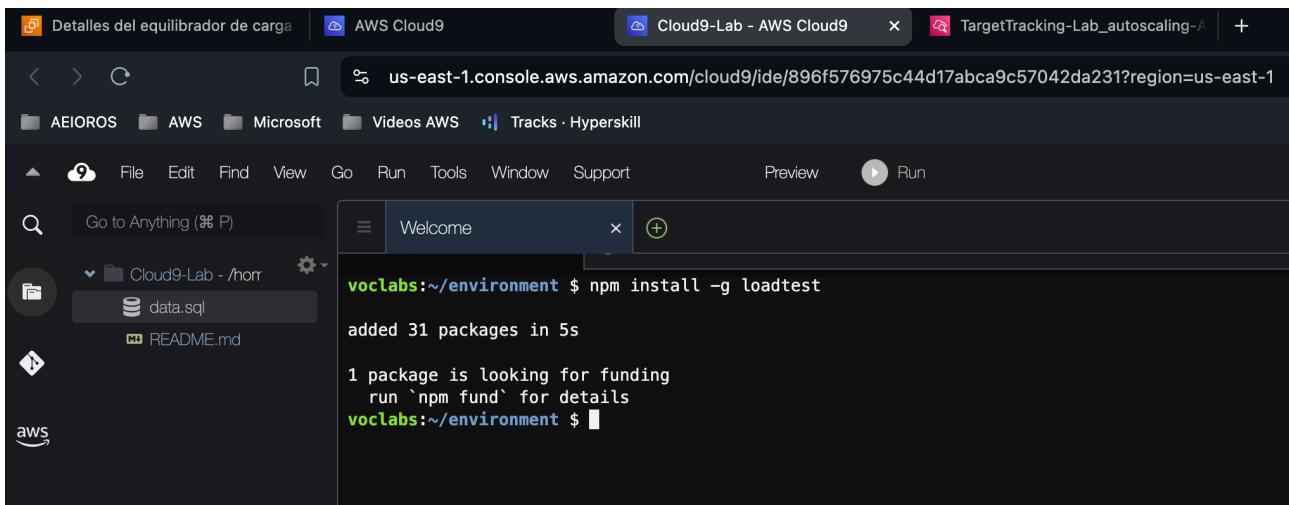
```
#Following command performs load testing on the given URL. replace the URL with
Loadbalancer (or Public IP of EC2 instance)
```

```
# Press ctrl +C to stop the script
```

```
loadtest --rps 1000 -c 500 -k http://<ALB_DNS_NAME>
```

Primero, accedemos a nuestro entorno de AWS Cloud9. En la terminal de Cloud9, instalamos el paquete loadtest utilizando el siguiente comando:

```
npm install -g loadtest
```



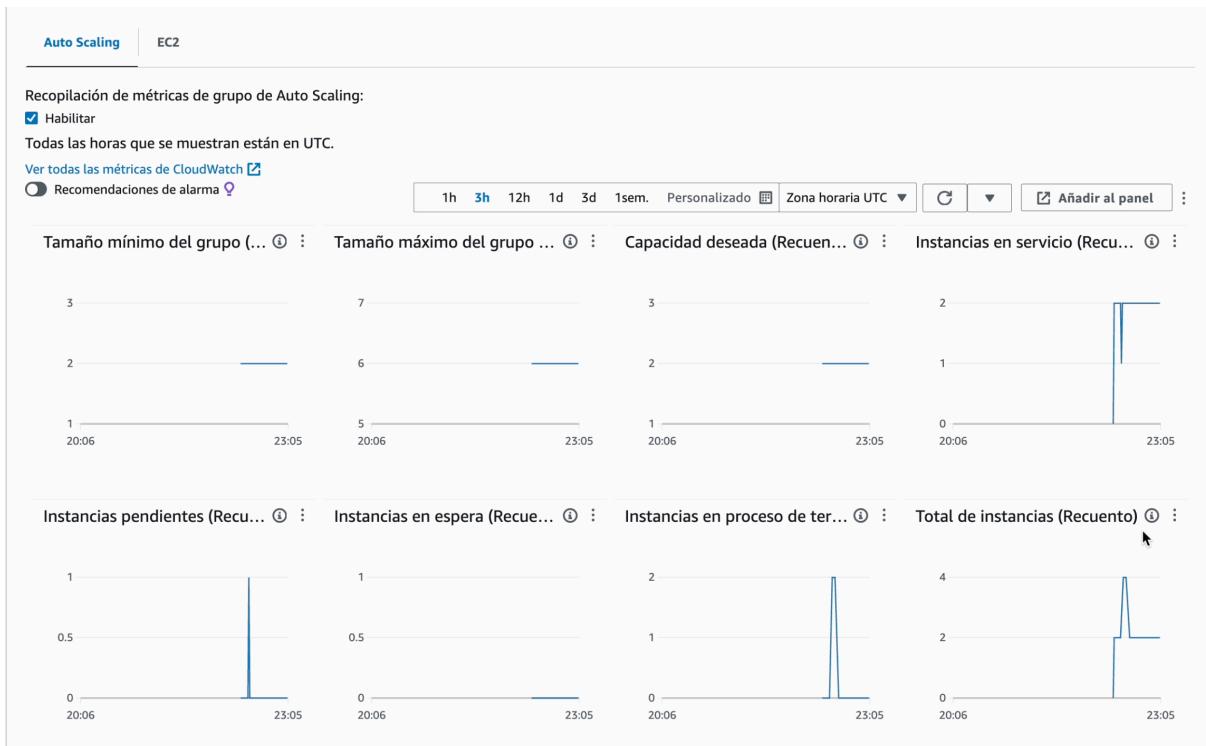
Este comando instala globalmente el paquete loadtest, una herramienta que utilizaremos para generar la carga en la aplicación web.

A continuación, ejecutamos el siguiente comando para iniciar la prueba de carga. Reemplaza <ALB\_DNS\_NAME> con el DNS del Application Load Balancer:

```
loadtest --rps 1000 -c 500 -k alb-webapp-1028279269.us-east-1.elb.amazonaws.com
```

Este comando enviará 1000 solicitudes por segundo a la aplicación con una concurrencia de 500 usuarios simultáneos. La opción -k permite mantener las conexiones vivas, lo que simula un tráfico más realista. Para detener el script, puedes presionar Ctrl + C.

Mientras la prueba de carga está en ejecución, accedemos a la consola de EC2 y navegamos a “Auto Scaling Groups”. Seleccionamos nuestro grupo de Auto Scaling y monitoreamos las métricas de utilización de la CPU y el número de instancias activas. Deberías observar que el número de instancias aumenta automáticamente en respuesta a la carga adicional, manteniendo la utilización de la CPU dentro del rango objetivo que configuraste anteriormente.



Después de que la prueba de carga haya terminado, revisa los resultados en la terminal de Cloud9 para evaluar el rendimiento de la aplicación. El comando loadtest te proporcionará estadísticas sobre las solicitudes, como el número de solicitudes completadas, la tasa de error, el tiempo de respuesta promedio, y más.

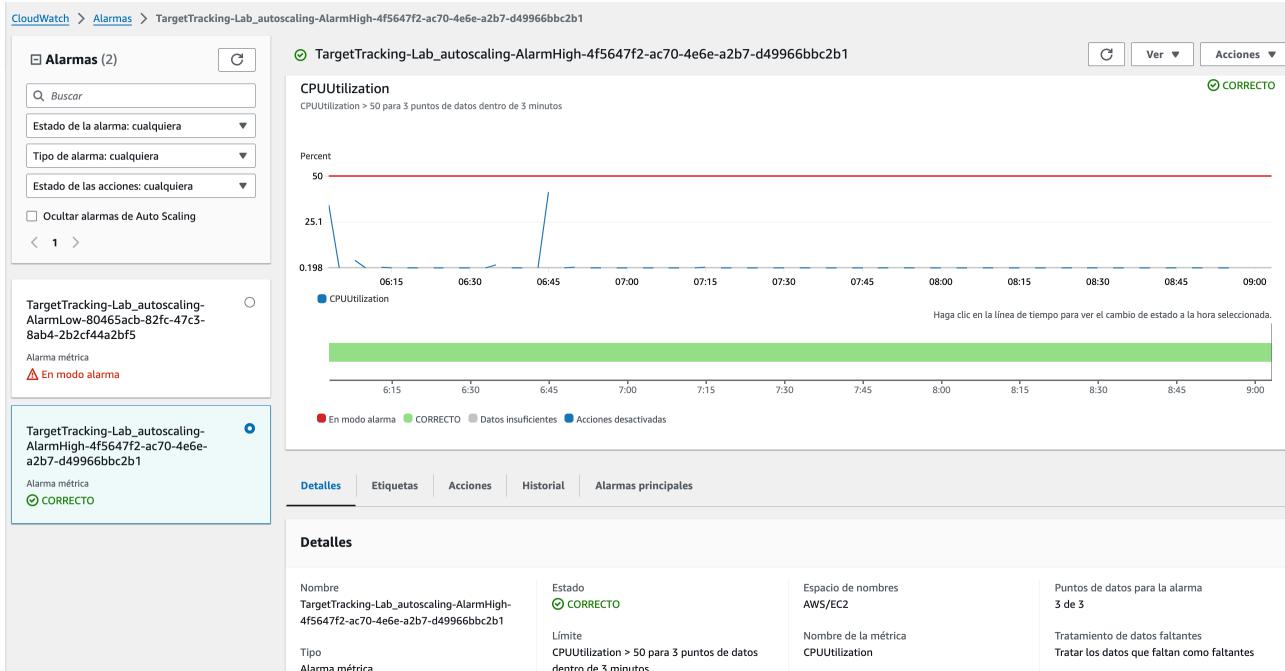
```

vclabs:~/environment $ loadtest --rps 1000 -c 500 -k http://alb-webapp-1028279269.us-east-1.elb.amazonaws.com/students
Requests: 1437, requests per second: 287, mean latency: 545.1 ms
Target URL: http://alb-webapp-1028279269.us-east-1.elb.amazonaws.com/students
Max time (s): 10
Target rps: 1000
Concurrent clients: 6711
Agent: keepalive
Completed requests: 3292
Total errors: 0
Total time: 10.002 s
Mean latency: 853.3 ms
Effective rps: 329

Percentage of requests served within a certain time
50%    496 ms
90%    723 ms
95%    5106 ms
99%    7709 ms
100%   9555 ms (longest request)

```

Además, revisamos los registros y métricas en la consola de CloudWatch para obtener información detallada sobre el comportamiento de la aplicación bajo carga. Esto nos permitirá identificar cualquier problema de rendimiento y asegurar que el escalado automático funcione según lo esperado.



## Evaluación del Cumplimiento de los Objetivos

- Funcionalidad:** La aplicación cumple con los requisitos funcionales, permitiendo ver, añadir, eliminar y modificar los registros de los estudiantes sin retrasos perceptibles bajo cargas normales.
- Balanceo de Carga:** La solución implementa un ALB que equilibra adecuadamente el tráfico de usuarios.
- Escalabilidad:** La configuración de Auto Scaling asegura que la aplicación pueda adaptarse a la demanda.
- Alta Disponibilidad:** La arquitectura, que incluye múltiples zonas de disponibilidad y un ALB, asegura un tiempo de inactividad limitado.
- Seguridad:** Se implementaron grupos de seguridad y AWS Secrets Manager para proteger los recursos y credenciales de la base de datos.
- Optimización de Costos:** La solución está diseñada para mantener los costos bajos mediante el uso de instancias y servicios de AWS eficientes.

- **Alto Rendimiento:** Las pruebas de carga confirmaron que la aplicación puede manejar grandes volúmenes de tráfico, aunque se identificaron áreas para optimización adicional.

El proyecto ha cumplido con éxito los objetivos planteados, diseñando e implementando una arquitectura de alta disponibilidad y escalabilidad en AWS para una aplicación web de registros de estudiantes. Se lograron configuraciones de balanceo de carga, autoescalado y seguridad robusta, asegurando que la aplicación pueda manejar eficientemente el tráfico esperado durante los períodos de mayor demanda.

## Conclusión

El proyecto de creación de una aplicación web de alta disponibilidad y escalabilidad en AWS ha sido un éxito, cumpliendo con todos los objetivos planteados. A través de este proyecto, hemos aprendido a diseñar e implementar una arquitectura robusta y eficiente utilizando una variedad de servicios de AWS, lo que nos ha permitido adquirir habilidades clave en la gestión y optimización de aplicaciones en la nube.

### Aprendizajes Clave

#### 1. *Diseño y Planificación de Arquitecturas en la Nube:*

Hemos aprendido a crear diagramas de arquitectura que representan la interacción entre varios servicios de AWS, como VPC, subredes, EC2, RDS, ALB y Auto Scaling. Este conocimiento es fundamental para diseñar soluciones escalables y de alta disponibilidad en cualquier entorno de nube.

#### 2. *Configuración y Gestión de Redes Virtuales:*

La configuración de una VPC con subredes públicas y privadas, gateways de Internet y tablas de enrutamiento nos ha permitido comprender cómo segmentar y asegurar el tráfico de red dentro de AWS. Estas habilidades son esenciales para la creación de entornos de red seguros y eficientes.

#### 3. *Despliegue y Gestión de Aplicaciones Web:*

Aprendimos a lanzar instancias EC2, instalar aplicaciones web y configurar bases de datos relacionales utilizando Amazon RDS. Además, la utilización de AWS Secrets Manager para gestionar credenciales de forma segura nos ha enseñado prácticas recomendadas de seguridad en la gestión de secretos.

#### 4. *Implementación de Balanceo de Carga y Auto Scaling:*

La configuración de un Application Load Balancer (ALB) y un grupo de Auto Scaling nos ha proporcionado conocimientos prácticos sobre cómo distribuir la carga de tráfico y escalar automáticamente los recursos según la demanda. Esto es importante para mantener el rendimiento y la disponibilidad de las aplicaciones en entornos de producción.

## *5. Pruebas de Rendimiento y Optimización:*

Realizar pruebas de carga con herramientas como loadtest nos ha permitido evaluar el rendimiento de la aplicación bajo condiciones de alta demanda. Aprendimos a monitorear y ajustar las configuraciones de Auto Scaling y a identificar áreas para optimización adicional, asegurando que la aplicación pueda manejar grandes volúmenes de tráfico de manera eficiente.

## **Aplicaciones Futuras**

Las habilidades y conocimientos adquiridos durante este proyecto son aplicables a una amplia variedad de escenarios y entornos. En el futuro, podremos aplicar estas técnicas y mejores prácticas en diferentes contextos, tales como:

1. **Desarrollo de Aplicaciones Empresariales:** Implementar arquitecturas escalables y de alta disponibilidad para aplicaciones críticas que requieren un rendimiento y una disponibilidad óptimos.
2. **Migración a la Nube:** Ayudar a las organizaciones a migrar sus aplicaciones y servicios a AWS, asegurando que las nuevas arquitecturas sean seguras, eficientes y escalables.
3. **Optimización de Costos:** Utilizar servicios de AWS de manera eficiente para mantener los costos bajos mientras se maximiza el rendimiento y la disponibilidad de las aplicaciones.
4. **Gestión de Seguridad en la Nube:** Implementar prácticas avanzadas de seguridad, como la gestión de secretos y la configuración de políticas de acceso, para proteger los recursos en la nube.

En conclusión, este proyecto no solo ha demostrado nuestra capacidad para construir una solución técnica sólida en AWS, sino que también ha ampliado significativamente nuestras competencias en la gestión de aplicaciones en la nube. Las habilidades adquiridas serán invaluables para abordar futuros desafíos y oportunidades en el ámbito de la informática en la nube.

© 2023, Amazon Web Services, Inc. o sus filiales. Todos los derechos reservados.